Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 57 (2015) 703 – 709

Procedia Computer Science

3rd International Conference on Recent Trends in Computing 2015 (ICRTC-2015)

# Efficient key management and cipher text generation using BCD coded parity bits

Rahul Ranjan[a], Debabala Swain[b], Bijay Paikaray[a,b,*]

[a]*Rahul Ranjan, Dept. of CSE, CUTM*
[b]*Debabala Swain, School of Computer Engeering*
[a,b,*]*Bijay Paikaray, Dept. of CSE, CUTM*

**Abstract**

In this paper we have presented a new symmetric encryption technique where a BCD converter, a four bit parity checker along with a sign function is used to generate the key sequence. In the next subsequent steps the input text and the key sequence are combined using the bitwise OR gate and NOT gate to produce the cipher text easily. The similar kinds of operations are performed in reverse succession to extract the original text from the cipher. Finally, by evaluating the obtained results against the AEC, DEC and BLOWFISH algorithms the competence of the proposed algorithm can be approximated.

*Keywords*- BCD Coded Parity checker, Symmetric Encryption, Execution time.

## 1. Introduction

In the past few decades, information technology has penetrated more and more into our society with an unpredicted impact on our daily life. However, the attribute which make digital information system so pretty make them particularly more susceptible to a broad range of exploitation. In Symmetric cryptology, the intended senders and receivers are discriminated from unintended ones by assuming that they are only aware of the *keys*. In Symmetric Encryption the problem of secrecy protection by using the shared secret key to transform the message in such a way that the sent information cannot be recovered without this key. A common encryption technique consists of two mathematical series of transformations, i.e., an encryption function, and a decryption function. Several symmetric encryption algorithms are proposed with different ability like, cryptographic complexity, time complexities rather considering their processing capability during execution.

*E-mail* debabala.swain@gmail.com

1877-0509 © 2015 The Authors. Published by Elsevier B.V. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).
Peer-review under responsibility of organizing committee of the 3rd International Conference on Recent Trends in Computing 2015 (ICRTC-2015)
doi:10.1016/j.procs.2015.07.455

In this paper we have proposed a symmetric encryption technique also represented its processing ability on different machines in which it runs files with different sizes. It has also flexible characteristics with varying block length, key length.

## 2. Literature Survey

### *2.1 LEA: Link Encryption Algorithm Proposed Stream Cipher Algorithm [1]*

It is a stream cipher algorithm, which uses an 8-bit ASCII character for encryption and decryption. These characters combine with the plain text by bitwise addition to produce the cipher character. The algorithm is offering highest security level achieved through the high nonlinear complexity and the complete re-initialization before every encryption process [1]. This algorithm forms the key sequence by combining the two different sequences, where the first one has probable long period and the second one is high complexity by using nonlinear functions. Moreover, both of these sequences are statistically flat. The good statistical properties of the first sequence are in principle that of Liner Feedback Shift Registers (LFSRs) with primitive characteristic polynomials to achieve maximal-length period. Also, those of the second part are achieved by using well distributed generated substitution boxes, confusion, and diffusion. The LEA encryption algorithm can be divided into a driving part and a combining part. The driving part consists of a set of maximum length Linear Feedback Shift Registers. It mainly governs the state sequence of the generator and is responsible for providing sequences of large periods and good statistics. The combining part is essentially nonlinear. It has the task to make the cipher stream generation to be mathematically complex.

### *2.2 Random Block Length Based Cryptosystem through Multiple Cascaded Permutation-Combinations and Chaining of Blocks [2]*

In this technique, a user inputs a key of minimum length eight byte and a randomly generated session keys are used to encrypt a source block of text. A 256 bit block is encoded in three steps by using three different sub-keys of sizes 128, 192 and 256 bits. The transitional text blocks are passed through the cipher block stage. Finally the wrapping technique works on the intermediate block to produce the cipher. It presents a multistage encryption technique with small space overhead. The three steps substitution, folding and permutation using multi-dimensional matrix variable may enhance its strength. Using multiple keys with varying lengths in different stages of the encoding, the encryption process becomes more secure.

### *2.3 AIDEA – A Processing Capacity Based Cryptographic Approach [3]*

It is a symmetric block cipher algorithm, which is more likely to be used as Data Encryption Standard (DES) with a higher key size. Like other algorithms it can be scaled up or scaled down. Though it is a strong algorithm as key size and design is concerned, but it is slow during implementations. Its hardware implementation can't be used in all computing systems. In the implementation it can execute with scaled down or scaled up block lengths and key sizes. So it rewards much of the time consumption during encryption and decryption processes in different configurations with different processing capabilities, which results a higher throughput during communication**.**

## 3. Proposed Technique

1. The proposed algorithm uses a BCD converter, 4-bit parity checker to generate a sequence.

2. This sequence will be used to generate a model of substitution technique. Thus the algorithm is considered to be a substitution algorithm which uses a single key to be shared by both the sender and receiver.
3. The cipher processes the input element continuously, producing output one element at a time. The new encryption algorithm is based on the concept of Poly alphabet cipher which is an improvement over mono alphabet.

### *3.1 Algorithm for generating the sequence*

1. Consider the sequence for 0 to 255 values for a source text of size 256 characters.
2. Convert each element of the sequence into BCD form.
3. Represent the values of step 2 into signed bits using even parity checker.
4. Represent the above signed digits in matrix form.
5. Add 1 to each element of output of step 4.
6. Convert each row into ternary form to generate the sequence.

It can be seen that to extract the original information from the coded text is highly impossible for the third person who is not aware of encryption keys and the method of coding. Even if the algorithm is known it is very difficult to break the code and generate key, given the strength of the algorithm .Thus given a short response time through internet communication, the algorithm is supposed to be safe.

### *3.2 Algorithm for Encryption*

1. Let $C_i$ be the plain text for i = 0 to 255.
2. Let the alpha-numerical equivalent $\mathbf{NE_i}$ of a character $\mathbf{C_i}$, as $\mathbf{NE_i = N_i - 26}$.
   Where, $\mathbf{N_i}$ = ASCII value of $C_i$ , for each i from 0 to 255.
3. Find the sum index $\mathbf{S_i = NE_i + K_i}$ , for each i from 0 to 255, where $\mathbf{K_i}$ is the $\mathbf{i^{th}}$ key generated in the sequence.
4. Find the 1's complement of $S_i$ using 8-bit NOT gate, let it be $SC_i$.
5. Find the cipher text for each input character as $\mathbf{CT_i}$ , where $\mathbf{CT_i}$ is the character equivalent of $\mathbf{SC_i}$, where i varies from 0 to 255.

### *3.3 Algorithm for Decryption*

1. Let $CT_i$ be the cipher text, where i=0 to 255..
2. Find the 1's complement equivalent binary code of $CT_i$as $CC_i$ .
3. Find the Difference index $\mathbf{D_i = CC_i - K_i}$ , for i= 0 to 255, where $\mathbf{K_i}$ is the $\mathbf{i^{th}}$ key generated in the sequence.
4. Find theNumeric Equivalent as $AN_i$, where $AN_i=D_i+26$.
5. Find $C_i$, where $C_i$ is the character equivalent of $AN_i$, for i = 0 to255.

## 4. Test Results and Performance Analysis

### *4.1 Implementation Details*

Algorithms were implemented using C programming code in Ubuntu 14.04 LTS 32 bit version on a 2.9 GHz Intel Pentium® processor using 2GB RAM to analyze their performance.

### *4.2 Complexity Analysis of Key Generation Algorithm*

In each step it performs n number of computations, so the total number of computations is in multiples of n. So the time complexity of the proposed technique can be expressed as **O (n)**.

### *4.3 Avalanche effect*

In Test-1, we have generated 256 key sequences using +1 for even parity and -1 for odd parity. In Test-2 we have we have substituted the parity values as -1 for even parity and +1 for odd parity. So there was a variation of 255 keys in the key sequence.

From the above test it was identified that a little variation in the parity values has a tremendous effect on the generated key sequence. Ultimately it produces a total new key set, which confirms about the high security feature of the proposed scheme. The Cipher text is dependent on the generated key sequence. So a small variant in the key sequence has a relative consequence on the cipher text, which provides maximum avalanche effect. It provides maximum strength and security to the algorithm. It can be easily revealed from the Table-1.

Table-1. Avalanche Test Analysis

| **Test No** | **No of modified Keys / Generated Keys** | **Original Text** | **Cipher Text** |
|---|---|---|---|
| 1 | 0/256 | Abcdef | ½¯°¿Ñ |
| 2 | 255/256 | Abcdef | ¯¿¯ |

Hence it is superior from other crypto algorithms. The complete Key sequence cannot be recovered; therefore the entire text cannot be retrieved.

### *4.4 Execution Time Analysis*

The nine text files of different sizes are used to carry out the tests, where an evaluation of three different algorithms AES, DES and Blowfish are performed. The experiments are conducted on the test system. The performance of these algorithms is evaluated based on parameters like, execution time, throughput and Avalanche effect.

Execution time is computed as the total time taken by the algorithm for encryption and decryption for a specific size of text data. Further the average execution time for individual algorithm is calculated.

Table-2.Execution Time Analysis of all four algorithms

| **INPUT SIZE (in KB)** | **EXECUTION TIME(in Sec)** | | | |
|---|---|---|---|---|
| | **DES** | **AES** | **BLOWFISH** | **PROPOSED** |
| **20** | 2 | 4 | 2 | 2.1 |
| **36** | 4 | 6 | 3 | 3.2 |

| | | | | |
|---|---|---|---|---|
| **45** | 5 | 8 | 4 | 3.6 |
| **59** | 7 | 11 | 6 | 4.5 |
| **69** | 9 | 13 | 7 | 5.1 |
| **137** | 17 | 26 | 14 | 8.5 |
| **158** | 20 | 30 | 16 | 9.6 |
| **166** | 21 | 31 | 17 | 10.2 |
| **191** | 24 | 36 | 19 | 11.6 |
| **232** | 30 | 44 | 24 | 13.1 |
| **Total Execution Time** | **139** | **209** | **112** | **71.5** |
| **Average Execution Time** | **13.9** | **20.9** | **11.2** | **7.15** |

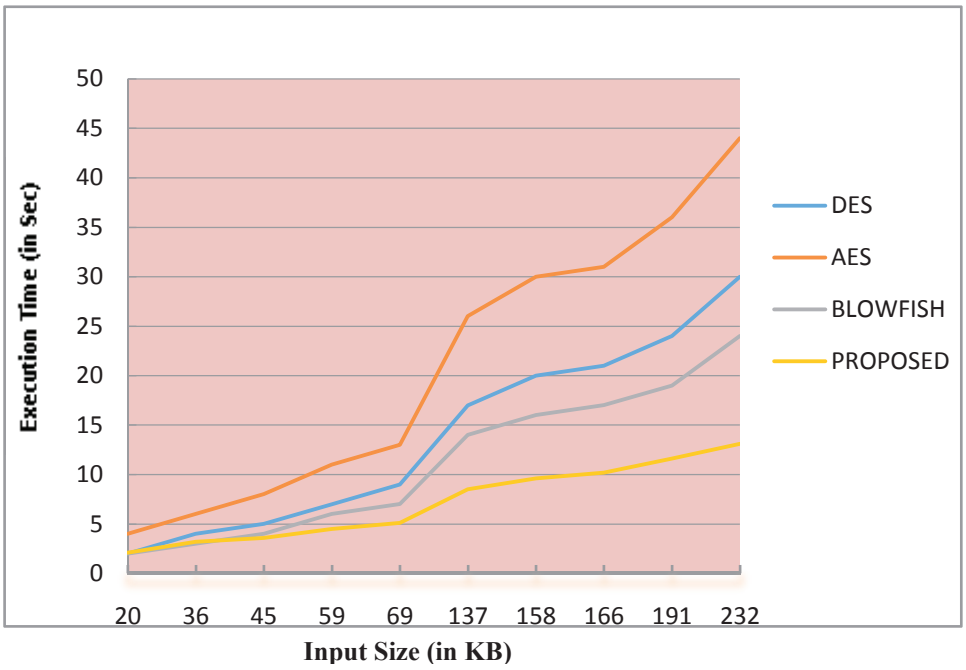


Fig.1.Execution time analysis between DES, AES, BLOWF1SHand PROPOSED algorithm

*4.5 Throughput Analysis*

Now the throughput of individual algorithm can be calculated by dividing the total data size in Kilo bytes to the execution time in seconds. Throughput of the algorithm is expressed in terms of kilobytes per second. As throughput is inversely proportionate to the consumed power of the algorithm, so it behaves as a major component of power management. The overall performance of all three algorithms along with the proposed algorithm is given in Table-3.

From the Table-3, it is clear that the proposed algorithm has improved performance than AES, DES and BLOWFISH. PROPOSED algorithm performs nearly 2 times faster than DES, 3 times faster than AES and 1.7 times faster than BLOWFISH algorithm. From the measured throughput figures AES is slowest and PROPOSED algorithm is fastest.

Table 3.Throughput Analysis of DES, AES, BLOWFISH, PROPOSED Algorithm

| **Algorithm** | **Throughout (in KB/sec)** |
|---|---|
| DES | 8.007 |
| AES | 5.325 |
| BLOWFISH | 9.937 |
| PRPOSED | 15.566 |

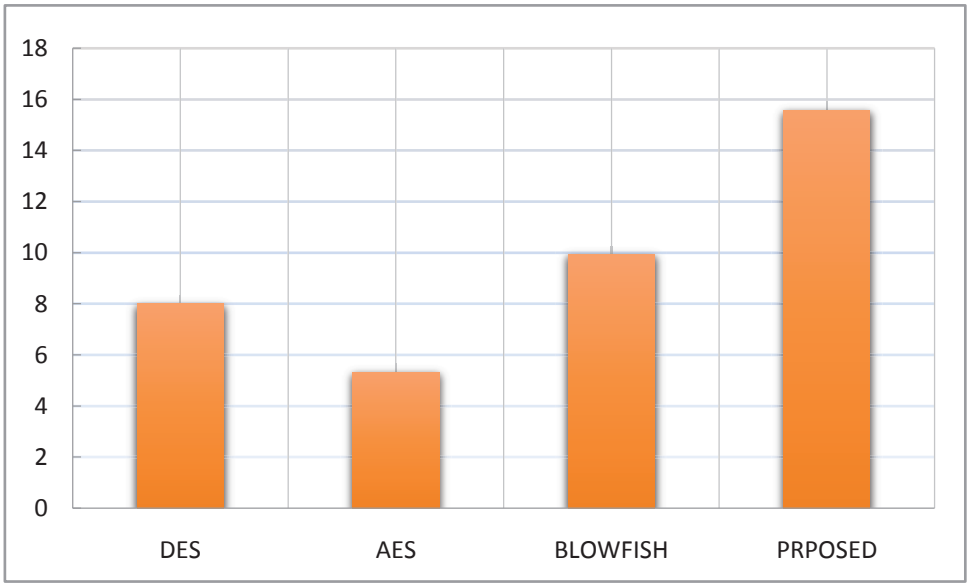


Fig 2.Throughput Analysis between DES, AES, BLOWFISH and PROPOSED algorithm

## 5. Conclusion

In information Technology where security is the major apprehension, there encryption algorithm plays a key role. In this paper we have implemented the foremost algorithms, like DES, AES and BLOWFISH and evaluated their performance along with the PROPESED algorithm by considering different parameters such as: execution time, throughput and Avalanche Effect. Our future work will direct more towards analysis of security

and performance issues of the PROPOSED algorithm using heterogeneous data types, like image encryption and video encryption.

## References

[1] LEA: Link Encryption Algorithm Proposed Stream Cipher Algorithm, Ain Shams Eng J (2014), http://dx.doi.org/10.1016/j.asej.2014.08.001,www.elsevier.com/locate/asej

[2] The Application of Hybrid Encryption Algorithm in Software Security, Fourth International Conference on Computational Intelligence and Communication Networks (CICN), 2012 http://ieeexplore.ieee.org/xpl/abstractAuthors.jsp?arnumber=6375216.

[3] Evaluation of Symmetric Encryption Algorithms for MANETs, Dec. 2010 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC).http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=5705754

[4]Baocang Wang, Qianhong Wu, Yupu Hu: A Knapsack Based Probabilistic Encryption Scheme, On Line March 2007.*dl.acm.org/citation.cfm?id=1279149.*

[5] Beth T. and Gollmann D. "Algorithm Engineering for Public Key Algorithms". IEEE Journal on Selected Areas in Communications; Vol. 7, No 4, PP. 458-466, http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=17708, July 2003.

[6] Performance Analysis of Encryption Algorithms for Information Security, 2013 International Conference on Circuits, Power and Computing Technologies (ICCPCT-2013), pp 840-844.